COMPUTER SCIENCE
SUMMER WORK

Welcome to Woodhouse College. We follow the AQA Computer Science Syllabus (7517), a linear course that culminates in 2 papers worth 40% each, and a large self-directed documented coursework worth 20%. The syllabus is not called "A-Level Python / C# / VB Programming" because *Computer Science is no more about computers, than literature is about spelling.* Though it is undoubtedly true that good programming knowledge is one of many essential skills. I would like to emphasize that equally as essential are a set of core skills you will develop over the course of your studies.

Thus, it makes sense that you have meaningful summer work tasks whose purpose is to allow you to hit the ground running (or at least walking) in September, to inspire you, to allow you to research, solve-problems and apply subject-specific skills. This document is filled with links to online material, where possible – attempt to view them. And then put these into practice, the reasons for practice are expressed succinctly:

> *"Not having heard something is not as good as having heard it; having heard it is not as good as having seen it; having seen it is not as good as knowing it; knowing it is not as good as putting it into practice." - Xunzi*

Tasks during the induction period:

1) **Organisational Skills (1.5 hrs)**

   Before you begin, watch this video on The Pomodoro Technique, a method that I personally found to be a useful way of maintaining focus. Organizational skills at college are crucial to success, and the sooner you can start independently learning the better. So start by looking at effective methods of Getting Things Done.

   Task:

   In addition to time-management – it would be beneficial to learn and apply the basics of a version control system like GitHub. To understand the basics, follow the Hello World Guide and then read GitHub for non-programmers. Once you have done this, your task is to start using GitHub to organize some of your previous files and projects and herewith work online. It will take time to formulate a method of your own that allows you to organize information you create so you can recall it quickly, the sooner you start the practice the better.

   Outcome:

   Learn to use GitHub to record the later exercises given. GitHub supports markdown – which is a lightweight and easy-to-use method for styling your text. It provides a perfect platform for writing rich notes with as little effort as possible. Click the link and practice editing documents within GitHub.

2) **Notetaking (1.5 hrs)**

   Cornell Notes serves as a good introduction to the skill of notetaking. And whilst not the only method of notetaking (see Chapter 10, The Study Skills Handbook 5th Ed. by Stella Cottrell for others), you can certainly use it as a base to build upon.

COMPUTER SCIENCE
SUMMER WORK

Watch How to Take Notes - Cornell Notes and then download the Cornell Notes Template. Print two copies of these (if you do not have a printer, just draw the lines on any paper – you just need the structure.)

Task:
Here is a 10-minute video of the Map of Computer Science that gives a brief overview of the subject. Pick any two of the topics presented there that most interest you – and then visit the CrashCourse Computer Science Playlist. Try and find the same content there [you may find that the topic is specialized and perhaps unavailable, select another topic.]

Write the title of the video and its topic in the top boxes (use a different sheet for each video)

In the main "Notes" section, write notes from the video. You can do this in any way you like, a suggestion might be to rewind slightly when the canvas changes, thinking carefully about what was important in the previous few minutes. Not everything is useful in the video, be selective in taking notes. Abbreviate where possible. *The most important part of learning is the recreation of what you have, written in your own language.* It is evidence of your understanding of what you have heard. It makes sense then, that you **minimize or never** copy or transcribe something word-for-word. Eliminate extra details and focus on communicating what is important.

Having watched the video, and recorded the notes, review them:

- Turn each part into a question in the section on the left. For example, the notes may say, "The value of the program counter is passed to the memory address register". The question then becomes, "Which register is the value of the program counter passed to?" Sometimes these questions are easy, and at times they are more difficult to write. There may also be more than one valid question and you will need to decide for yourself which are the most appropriate questions for revision. Throughout the course we will see examples of AQA questioning. For now, use your own judgement.
- Pull out all the key words, research their definitions in the notes, and list them in the bottom section.
- Finally, look through the AQA Syllabus and add the relevant index in the topic title.

Outcome:
You have notes on 2 subjects of interest. There's relevant condensed information there. The notes are easily indexed against the syllabus and can be filed in an appropriate section in your folder. You have a sense of the important sections and a relevant list of questions to consider later when you revise. You have a list of the relevant "Tier 3" specialized terminology that you can apply to answering exam questions later.

This skill can also be applied when taking notes from reading materials in a textbook or online. Read a paragraph, look away, then ask yourself what the paragraph meant. That ought to be your note.

3) Contextualisation (1.5 hrs)

COMPUTER SCIENCE
SUMMER WORK

In the exams, you will be presented with a scenario and some idea of the general theoretical area of concern in Computer Science. The scenario may be totally unfamiliar to you, and simply regurgitating what you know on the topic, without contextualising the answer – leads to low marks.

Task:
Watch this Ted Talk on how computers learn to recognise objects instantly. After watching this video, **discuss the benefits, limitations, and risks of the subject in the context of:**

- Travel and Tourism
- Gaming and Entertainment
- Education and Learning
- Transport and Navigation
- Medicine and Healthcare

Outcome:
Record the answers on a page on your GitHub. For each given context – you will write a short paragraph of the benefits, limits, and risks of fast image recognition in the context presented.

4) Abstract Thinking (30 mins)

Read the 5 strategies for generating new insights. The ability to think abstractly, is the ability to remove unnecessary detail from a topic, representation, or idea. The term used is abstraction – that is *a process of omitting all individuating features and retaining only what is common to all of a set of resembling particulars*.

Computers operate independently of the concrete world. Programs written for machines that interact with people, or things – are therefore written to work within a model of the real world. The model is an abstract representation that removes irrelevant details keeping only what is necessary. A critical skill you should develop, is the ability to apply abstraction.

You deal with abstractions all the time. Pushing the lever down on a toaster involves an abstraction that masks the details associated with turning the heating element on, starting a timer, detecting the temperature of the surface of the slice, and so on. From your perspective all you need to know is how to use the toaster, you need to provide the toast and switch it on. All the details of how hot the toaster should get, when it should pop up, setting the timer, and so on are unnecessary to you.

Task:
A palindrome is a word that reads identically backward or forward. Consider the word "RACECAR", what steps would you take to identify whether it was a palindrome? Write these down using concrete terms. Do the same for another word "DEFIED".

Given the concrete examples, consider the generalized problem - informally describe an algorithm in English, that can identify any palindromes.

Outcome:

In the course of thinking about the algorithm – you would replace the text sequence with an index. The index is the beginning of an abstraction. You would create a flow and probably repeat the process …. Fleshing out an algorithm. You would refine this until it becomes trivial to write the program.

Research:
What other examples are there of abstractions in Computer Science?

5) Programming Skills – Algorithms (4+ hrs):

Register with Sololearn and Repl.It . SoloLearn is an online learning platform for programming. Repl.it is an online platform that allows you to code without the need to install a programming development environment on your machine.

On Sololearn, select the Python Learning course, and complete modules 1, 2 and 3 that cover fundamentals of programming. Throughout the materials, Sololearn provides you with a button that runs the example code provided - however, rather than running the example programs written in Python on Sololearn - type the code out in Repl and run it there. The act of typing these examples is Active Learning.

Following that - you are to code 2 programs. It may be a good idea to try something simpler before moving on to the tasks below.

Task 1:
Create a **lottery number generator**. This must generate 6 random numbers between 1 and 49 inclusive, and a bonus ball. The numbers must not repeat in any draw. Feel free to "prettify" the text output.

Extension:
If you found the above quite simple, you can add the extension task. The lottery is an example of a permutation without repetition. Your extension task is to get an input from the user containing all 7 numbers. If the input is valid, show the user which of their numbers were correct, and return the probability of him having chosen that combination of numbers.

Task 2:
The second application is a word-opposites quiz game to help young students practice language. The program must randomly select **two** different pairs of words from either of the lists below, and then display a question based on the selection. You must ensure that you are not selecting the same synonym-antonym pair i.e. repeating the question.

Table 1Word lists

hot, summer, hard, dry, simple,	cold, winter, soft, wet, complex,
light, weak, male, sad, win, small,	darkness, strong, female, happy,
ignore, buy, succeed, reject,	lose, big, pay attention, sell, fail,
prevent, exclude	accept, allow, include

The quiz takes the following form:
(* assuming the words hot and happy were randomly selected.)

COMPUTER SCIENCE
SUMMER WORK

"Q1: Hot is to cold, as happy is to …. ?"
Answer: sad ← user types this.
"Q2: Ignore is to pay attention, as darkness is to …. ?"
Answer: …..
The program will ask 10 questions. Ensure that each of the questions would not give away the previous answers:

e.g.
"Q1: Hot is to cold, as happy is to …. ?"
"Q2: Ignore is to pay attention, as Hot is to …. ?" ← the answer was printed in Q1! You need to avoid this scenario.

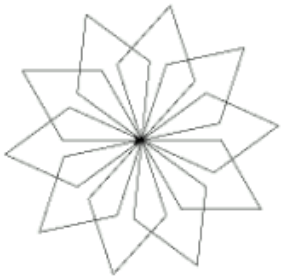
Return the score to the student and show the correct answers for those that were wrongly typed.

Extension:
This is a **much harder** extension, but if you feel like a challenge then read on. Extend the program to read the wordlists from a file. The file is available here, look at the structure of the document – you are required to parse the text and extract the contents. It is presented as KEY, SYNONYM_LIST, ANTONYM_LIST. Select any two random keys, and have the program quiz the student as before. Maintain a total running score as the students answer the questions.

Note: The given wordlist is interesting, there are entries in the text that are enclosed in square brackets, with a reference [See XXXXX]. For extra kudos points, can you derive the link and present the contents of the link as an extension to the relevant KEY entry?

[PS: I appreciate the above is difficult, this is more a stretch challenge. If the code is difficult, consider writing "pseudocode" (i.e. a list of instructions in plain English that outlines the method you would use to tackle that.)

6) Logical Thinking – Pattern Recognition (1 hr):



Repl provides another Python environment that has partial support for a graphics module called "Turtle Graphics". Create a new "Python with Turtle" repl and then watch this complete Python Turtle graphics overview.

Task:
Write a series of step-by-step instructions to reproduce the above shape.

COMPUTER SCIENCE
SUMMER WORK

Outcome:
In the context of the title – you can see that the flower-shape involves a repetition of the same "petal" a number of times, about a centralised point. Thus, if you knew how to generate a single petal, and to rotate around the same corner, then the task of repeating it a number of times becomes trivial. You have recognised a fundamental pattern. Pattern recognition isn't just graphical – it also involves more complicated thinking. You did this earlier when you fleshed out the details of an algorithm identifying palindromes. When you identify commonalities in experiences, you can extract those patterns allowing you to create a general idea of the problem and how to solve it.

These concepts also extend to programming "patterns", its an area called "design patterns".

7) Research and Critical Thinking. (2.5 hrs)

"I know that I know nothing." – Plato on Socrates

Our brains tend to look for shortcuts and easy answers. It has adapted over time to make quick judgments on a wide variety of confusing events in the world. This is the reason why optical illusions work. It is also the reason why subjective opinion and bias is inevitable, and that we will leap to the wrong conclusion. To combat this, one must develop critical thinking skills, it is a skill that is considered a cornerstone of academic maturity.

Please read this Guide to Critical Thinking and these Personal Strategies for Critical Thinking. (For further research, read Chapter 12 of the Study Skills Handbook 5th Ed.)

Computer Science comprises a multitude of disciplines and topics, some of which are only touched upon briefly at A-Level and some not at all. Watch these 9 examples of specification gaming.

Task:
A statement is posed:

"AI doesn't have to be evil to destroy humanity – if AI has a goal and humanity just happens in the way, it will destroy humanity as a matter of course without even thinking about it, no hard feelings." – Elon Musk

On no more than 1 side of A4 – research the statement and evaluate it critically. Go beyond the description, analyse any data, look at different perspectives, justify your position and reach a conclusion. Page 8 of this guide lists some questions you might want to consider.

Outcome:

"I keep six honest serving-men: (they taught me all I knew) their names are What and Where and When and How and Why and Who." - Kipling

Critical thinking is a commitment and disposition to subject all thinking to rigorous critique. At this stage, you were able to overcome subjective opinions and respond to a claim using academic rigour that would stand the test of scrutiny.

COMPUTER SCIENCE
SUMMER WORK

It is with hope, that you have practised and continue to develop the skills listed here. There has been little covered in terms of the theory of Computer Science. The skills required are transferable and essential in all subjects, but particularly in CS. Keep a record of the work you have carried out on GitHub. We will discuss the work should classes commence in September.

COMPUTER SCIENCE
SUMMER WORK

Name: Sergi **Date:** 30/8/2020	**Computer Science Summer Work, 2020** **Checklist**

Organisational

I can apply the Pomodoro technique to stay focused in manageable timeslots. **Y / N**

I understand the core principles behind GTD **Y / N**

I have gained practical, first-hand experience of GitHub and set up an account there. **[Evidence of Account]** **Y / N**

I have created notes on GitHub using Markdown. **[Evidence of Markdown related notes]** **Y / N**

Notetaking

I understand the methods used for Cornel Notes. **Y / N**

I appreciate the wide-ranging scope of studying CS. **Y / N**

I have prepared a set of notes on the following CrashCourse video: Machine Learning and AI which is in section 4. _ . _ of the AQA specification. **[Fill in the detail above, provide Evidence of Note]** **Y / N**

Not in AQA spec

I have prepared a set of notes on the following CrashCourse video: Future of computing which is in section 4. _ . _ of the AQA specification. **[Fill in the detail above, provide Evidence of Note]** **Y / N**

Not in AQA spec

Contexualisation

I have a page on GitHub with 5 paragraphs featuring the concept of fast image recognition in the context of tourism, entertainment, education, transport and healthcare. **[Evidence on GitHub Page]** **Y / N**

Abstraction

I have a page on GitHub that outlines:

1) the process of checking “RACECAR” is a palindrome **Y / N**
2) the process of checking “DEFIED” is not a palindrome **Y / N**
3) the process of checking if any word is a palindrome **Y / N**

[Evidence on GitHub page]

I have considered other areas of abstraction in CS **Y / N**

Programming

I have completed module 1 on Sololearn, running code on Repl **Y / N**
I have completed module 2 on Sololearn, running code on Repl **Y / N**
I have completed module 3 on Sololearn, running code on Repl **Y / N**

COMPUTER SCIENCE
SUMMER WORK

Task 1: **[Evidence of code on GitHub, attached through Repl]**	
I have a working lottery number generator.	O
I have a partially working lottery number generator.	O
I have generated 7 random numbers	Y / N
I ensure no repetitions were present	Y / N
I have formatted the text-output and added flair.	Y / N
Extension:	
I have done the optional extension task.	O
I have partially done the optional extension task.	O
I have validated the user input	Y / N
I have shown the correct combination of input	Y / N
I have calculated the probability	Y / N
Task 2: **[Evidence of code on GitHub, attached through Repl]**	
I have a working word-opposites quiz game.	O
I have a partially working word-opposites quiz game.	O
I have repeated the questioning 10 times.	Y / N
I have tallied a score.	Y / N
I have ensured no duplicates arise.	Y / N
I have printed the correct answers.	Y / N
Extension:	
I have done the optional extension task.	O
I have partially done the optional extension task.	O
I have parsed the contents of the file deriving Key/Value pairs.	Y / N
I have extended the mapping of key-value pairs by handling references.	Y / N
I did not do this programmatically, but I have documented the set of instructions that may work.	or Y / N
I found another way to do this.	or 2Y / N
Pattern Recognition	
I understand Python Turtle graphics	Y / N
I have a GitHub page outlining the step-by-step instructions to reproduce the given shape. [Evidence on GitHub]	Y / N
Research & Critical Thinking	
I appreciate how to apply methods of Critical Thinking through questioning.	Y / N
I can research topics using robust sources.	Y / N
I have written a single A4 page critically evaluating the statement given.	Y / N

COMPUTER SCIENCE
SUMMER WORK

Please tally up all "Y" marked answers above; write the total here: 32 **/33**